REMARKS

Claims 1-26 and 40-67 are currently pending in this application. By this response to the non-final Office Action dated March 19, 2009, claims 1, 12-26, 40, 50, and 54-67 are amended,. Support for the amendments is found in the specification, including the claims, as originally filed. No new matter has been introduced. Favorable reconsideration of the application in light of the foregoing amendments and following comments is respectfully solicited.

Claims 12 and 25 were objected to in view of minor informalities.

Claims 1, 2, 4-8, 11-15, 17-21, 24-26, 40, 41, 43, 46, 50, 54, 55, 57, 60, and 64 were rejected under 35 U.S.C. § 102(b) as being anticipated by Blelloch (*Vector Models for Data-Parallel Computing*).

Claims 1, 11, 14, and 24 were rejected under 35 U.S.C. § 102(e) as being anticipated by Lee (U.S. Patent No. 6,381,690).

Claims 12 and 25 were rejected under 35 U.S.C. § 103(a) as being unpatentable over Lee in view of Matsuura (U.S. Patent No. 4,725,973).

Claims 3, 9, 10, 16, 22, 23, 42, 44, 45, 47-49, 51-53, 56, 58, 59, 61-63, and 65-67 were rejected under 35 U.S.C. § 103(a) as being unpatentable over Blelloch in view of *In re Rose*, 105 USPQ 237 (CCPA 1955).

Applicants respectfully traverse.

I. PRIORITY

Section 2of the Office Action questions the priority claim of the present application. According to the Office Action, the disclosure of U.S. Patent Application No. 09/169,963 (now issued as U.S. Patent No. 6,006,318) (referred to herein as the '318 patent), from which the

present application claims priority, fails to provide adequate support for the present claims to satisfy the requirements under 35 U.S.C. § 112, paragraph 1. In particular, the Office Action at page 3, first paragraph, alleges that the disclosure of the '318 patent does not describe the operation of an instruction that allows fields of a data selection operand to independently select data elements, using claim 1 as an example. This limitation of claim 1 is reproduced below (as currently amended):

> **the data selection operand comprising a plurality of fields each independently selecting one of the plurality of data elements; and providing in parallel the data elements selected by the fields to respective predetermined positions in a catenated result, wherein the predetermined positions are in the same order as the fields of the data selection operand.**

Applicants respectfully submit that the disclosure of the '318 does provide adequate support for the present claims, including the above-quoted limitation of claim 1 relating to an instruction that allows fields of a data selection operand to independently select data elements. Specifically, the disclosure of the '318 patent includes a microfiche appendix that contains a detailed description of the operation of the G.SELECT.8 instruction, which is also mentioned in the specification of the '318 patent. The G.SELECT.8 instruction is an example of an instruction that allows fields of a data selection operand to independently select data elements, in accordance with the limitations of claim 1.

A detailed discussion is provided below relating to (1) the microfiche appendix, (2) the description of the G.SELECT.8 instruction, and (3) the manner in which the '318 disclosure meets the enablement and written description requirements under § 112, to support the priority claim of the presently claimed invention.

A. Microfiche appendix

The disclosure of the '318 patent includes a microfiche appendix, which was filed as part of the original application (U.S. Patent Application No. 09/169,936) for the '318 patent. As the specification of the '318 patent states:

A Microfiche Appendix consisting of 4 sheets (387 total frames) of microfiche is included in this application. The Microfiche Appendix contains material that is subject to copyright protection. The copyright owner has no objection to the facsimile reproduction by any one of the Microfiche Appendix, as it appears in the Patent and Trademark Office patent files or records, but otherwise reserves all copy rights whatsoever. '318 patent, col. 1, liens 10-17.

The microfiche appendix includes a version of the Terpischore System Architecture manual. This manual contains a detailed description of the entire instruction set of the Euterpe microprocessor, which represents an embodiment of the claimed invention. Each instruction in the instruction set is described in pseudo code, which fully defines the operation of the instruction, down to the bit level. One such instruction is the G.SELECT.8 instruction. The following section discusses the detailed description of the G.SELECT.8 instruction as found in the microfiche appendix.

B. Detailed description of the G.SELECT.8 instruction

The microfiche appendix includes a detailed description of the G.SELECT.8 instruction that fully defines the operation of this instruction. Relevant portions of the microfiche appendix are reproduced below and highlighted for ease of review:

Group Ternary

These operations perform calculations with three general register values, placing the result in a fourth general register.

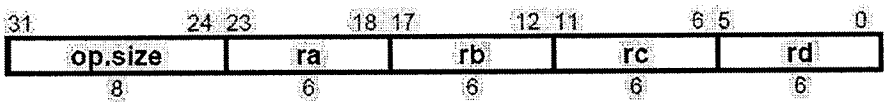
Operation codes

G.8MUX	Group 8-way multiplex
G.EXTRACT.128	Group extract hexlet
G.GF.MUL.8	Group Galios field multiplex byte
G.MULADD.1[25]	Group signed multiply bits and add pecks
G.MULADD.2	Group signed multiply pecks and add nibbles
G.MULADD.4	Group signed multiply nibbles and add bytes
G.MULADD.8	Group signed multiply bytes and add doublets
G.MULADD.16	Group signed multiply doublets and add quadlets
G.MULADD.32	Group signed multiply quadlets and add octlets
G.MULADD.64	Group signed multiply octlets and add hexlets
G.MUX	Group multiplex
G.SELECT.8	Group select bytes
G.TRANSPOSE.8MUX	Group transpose and 8-way multiplex
G.U.MULADD.2	Group unsigned multiply pecks and add nibbles
G.U.MULADD.4	Group unsigned multiply nibbles and add bytes
G.U.MULADD.8	Group unsigned multiply bytes and add doublets
G.U.MULADD.16	Group unsigned multiply doublets and add quadlets
G.U.MULADD.32	Group unsigned multiply quadlets and add octlets
G.U.MULADD.64	Group unsigned multiply octlets and add hexlets

class	op	size
extract	EXTRACT	128
signed multiply and add	MULADD	1 2 4 8 16 32 64
unsigned multiply and add	U.MULADD	2 4 8 16 32 64
Galois field multiply	GF.MUL	8
multiplex	MUX 8MUX TRANSPOSE.8MUX	NONE
select	SELECT	8

Format

G.op.size rd=ra,rb,rc



Description

The contents of registers or register pairs specified by ra, rb, and rc are fetched. The specified operation is performed on these operands. The result is placed into the register pair specified by rd.

Definition

```
def c ← PolyMultiply(size,a,b) as
      p[0] ← 0^2*size
      for k ← 0 to size-1
            p[k+1] ← p[k] ^ a_k?(0^size-k || b || 0^k) : 0^2*size
      endfor
      c ← p[size]
enddef

def c ← PolyResidue(size,a,b) as
      p
enddef

def GroupTernary(op,size,ra,rb,rc,rd) as
    case op of
          G.MUX, G.GF.MUL:
                a ← RegRead(ra, 128)
                b ← RegRead(rb, 128)
                c ← RegRead(rc, 128)
          G.EXTRACT, G.8MUX, G.TRANSPOSE.8MUX:
                a ← RegRead(ra, 128)
                b ← RegRead(rb, 128)
                c ← RegRead(rc, 64)
          G.MULADD:

          G.U.MULADD:
                a ← RegRead(ra, 64)
                b ← RegRead(rb, 64)
                c ← RegRead(rc, 128)
          G.SELECT:
                a ← RegRead(ra, 64)
                b ← RegRead(rb, 64)
                c ← RegRead(rc, 64)
    endcase
    case op of
          G.GF.MUL:
                for i ← 0 to 128-size by size
                      d_size-1+i..i ← PolyResidue(8,PolyMul(8,a_size-1+i..i, b_size-1+i..i),c_size-1+i..i)
```

```
            endfor
        G.MUX:
            d ← (b and a) or (c andnot a)
        G.8MUX:
            for i ← 0 to 127
                d_i ← a_(i_6..3 || c_i&63 || b_(i&63)+64 || b_i&63)
            endfor
        G.TRANSPOSE.8MUX:
            for i ← 0 to 127
                t_i ← a_(i_6..6 || i_2..0 || i_5..3)
            endfor
            for i ← 0 to 127
                d_i ← t_(i_6..3 || c_i&63 || b_(i&63)+64 || b_i&63)
            endfor
        G.EXTRACT:
            d ← (a || b)_(c&127)+127..(c&127)
        G.MULADD:
            for i ← 0 to 64-size by size
                d_2*(i+size)-1..2*i ← c_2*(i+size)-1..2*i +
                        (a_size-1^size || a_size-1+i..i) * (b_size-1^size || b_size-1+i..i)
            endfor
        G.U.MULADD:
            for i ← 0 to 64-size by size
                d_2*(i+size)-1..2*i ← c_2*(i+size)-1..2*i +
                        (0^size || a_size-1+i..i) * (0^size || b_size-1+i..i)
            endfor
        G.SELECT:
            ab ← a || b
            for i ← 0 to 15
                j ← c_4*i+3..4*i
                c_8*i+7..8*i ← ab_8*j+7..8*j
            endfor
    endcase
    RegWrite(rd, 128, d)
enddef
```

The portions of the microfiche appendix reproduced here describe in detail, down to the bit level, the exact operation of the G.SELECT.8 instruction, and would be easily understood by one of ordinary skill in the art. As can be seen, the G.SELECT.8 instruction takes its operands from three 64-bit registers, r_a, r_b, and r_c. First, the 64-bit operand from register r_a and the 64-bit operand from register r_b are appended together to form a 128-bit data source, which is represented by a variable "ab" (ab ← a || b). The 128-bit data source can be seen as sixteen 8-bit bytes. Next, the 64-bit operand from register r_c is parsed into sixteen 4-bit data selection

operands, which are represented by a variable "j" ($j \leftarrow c_{4*i+3..4*i}$, where the value of i ranges from 0 to 15). Each of these 4-bit data selection operands independently selects a byte of data from the 128-bit data source. Each 4-bit data selection operand can freely select any one of the sixteen bytes of data found in the data source. There is no restriction that the 4-bit data selection operands must be different from one another. In other words, multiple 4-bit data selection operands may select the same byte of data from the 128-bit data source (in that case, the selected byte of data would be repeated in the result). Finally, the bytes of data selected by the 4-bit data selection operands are provided to respective predetermined positions in the result ($c_{8*i+7..8*i} \leftarrow ab_{8*j+7..8*j}$). These predetermined positions appear in the result in the same order as the 4-bit data selection operands.

C. Enablement and written description requirements met under § 112

Applicants respectfully submit that the detailed description of the G.SELECT.8 instruction found in the microfiche appendix provides proper support for the presently claimed invention, to satisfy both the enablement and written description requirements under 35 U.S.C. § 112, paragraph 1. The G.SELECT.8 instruction is an example of an instruction that allows fields of a data selection operand to independently select data elements, in accordance with the limitations of claim 1. As can be seen from the discussion above, the pseudo code disclosed in the microfiche appendix describes in detail, down to the bit level, the exact operation of the G.SELECT.8 instruction. Given this level of detail, one of ordinary skill in the art would be able to practice the claimed invention without undue experimentation, or any experimentation at all for that matter. The pseudo code disclosed in the microfiche appendix also provides written description of the claimed invention, by fully describing an embodiment of the invention.

A claim chart is presented below showing, for each limitation of claim 1, the corresponding support for that limitation as found in the '318 disclosure (including the '318 patent specification and the microfiche appendix).[1]

Claim Limitation	Support in '318 Disclosure
1. A method of processing data in a single programmable processor, the method comprising:	"A general purpose, programmable media processor for processing and transmitting a media data stream of audio, video, radio, graphics, encryption, authentication, and networking information in real-time." '318 patent, abstract, paragraph 1. "Terpsichore's Euterpe processor performs integer, floating point, and signal processing operations at data rates up to 512 bits (i.e., up to four 128-bit operand groups) per instruction. The instruction set design carries the concept of streamlining beyond Reduced Instruction Set Computer (RISC) architectures, since it targets implementations that issue several instructions per machine cycle." '318 appendix, p. 11, para. 2.
decoding a single instruction for selectively arranging data, specifying a data selection operand and a first and a second register each having a register width,	"G.SELECT.8" '318 patent, cols. 13 and 14, Table 1. "G.SELECT.8 – Group Select Bytes" *See* definition of "G.SELECT.8" instruction, '318 microfiche appendix, pp. 126-128. "The contents of registers or register pairs specified by ra, rb, and rc are fetched. The specified operation is performed on these operands. The result is placed into the register pair specified by rd." *Id.* G.SELECT: a ← RegRead(ra, 64) b ← RegRead(rb, 64) c ← RegRead(rc, 64) *Id.*

[1] This claim chart focuses on claim 1, because the Office Action specifically pointed to claim 1 in the discussion regarding support under § 112. Similar claim charts can be prepared for each of the remaining claims of the present application, if the Patent Office deems such additional claim charts necessary.

	Here, the data selection operand is specified as the contents of register r_c. The first and second operand are specified as registers r_a and r_b, respectively. Registers r_a and r_b each have a register width of 64 bits.
the first and second registers providing a plurality of data elements each having an elemental width smaller than the register width,	Registers r_a and r_b together provide a 128-bit data source, which consists of sixteen 8-bit data elements (sixteen bytes). Each data element has an elemental width of 8 bits, which is smaller than the register width, which is 64 bits. The 128-bit data source is placed in variable "ab" (ab ← a || b). *See* definition of "G.SELECT.8" instruction, '318 microfiche appendix, pp. 126-128.
the data selection operand comprising a plurality of fields each independently selecting one of the plurality of data elements; and providing in parallel the data elements selected by the fields to respective predetermined positions in a catenated result, wherein the predetermined positions are in the same order as the fields of the data selection operand.	The 64-bit operand from register r_c is parsed into sixteen 4-bit data selection operands, and placed into variable "j" ($j \leftarrow c_{4*i+3..4*i}$, where the value of i ranges from 0 to 15). Each of these 4-bit data selection operands independently selects a byte of data from the 128-bit data source. The bytes of data selected by the 4-bit data selection operands are provided to respective predetermined positions in the result ($c_{8*i+7..8*i} \leftarrow ab_{8*j+7..8*j}$). These predetermined positions appear in the result in the same order as the 4-bit data selection operands. *See* definition of "G.SELECT.8" instruction, '318 microfiche appendix, pp. 126-128.

Accordingly, it is believed that the priority claim of the present application is properly made. The present application claims priority from the application of the '318 patent. The microfiche appendix included as part of the original disclosure of the '318 patent provides a detailed description of the operation of the G.SELECT.8 instruction, which fully supports the presently claimed invention to satisfy both the enablement and written description requirements under 35 U.S.C. § 112. As such, confirmation of the priority claim of the present application is respectfully requested.

II. CLAIM OBJECTIONS

In section 5 of the Office Action, claims 12 and 25 were objected to in view of minor informalities. Applicants thank the Examiner for providing suggestions for amending these claims to resolve the informalities. Claims 12 and 25 have been amended in accordance with the Examiner's suggestions, in order to overcome the objections.

III. REJECTIONS UNDER 35 U.S.C. §§ 102 AND 103

A. Blelloch

In section 7 of the Office Action, claims 1, 2, 4-8, 11-15, 17-21, 24-26, 40, 41, 43, 46, 50, 54, 55, 57, 60, and 64 were rejected under 35 U.S.C. § 102(b) as being anticipated by Blelloch (Vector Models for Data-Parallel Computing). Applicants respectfully submit that the claims as currently amended distinguish over Blelloch for at least two important reasons. First, Blelloch fails to disclose a data selection operand comprising a plurality of fields each independently selecting one of a plurality of data elements. Second, Blelloch fails to disclose, in a single programmable processor, providing in parallel the data elements selected by the fields to respective predetermined positions in a catenated result. These reasons are discussed in detail below.

1. Blelloch fails to disclose a data selection operand comprising a plurality of fields each independently selecting one of a plurality of data elements

Blelloch fails to disclose a plurality of fields each independently selecting one of the plurality of data elements. Claim 1 recites, in part:

> **decoding a single instruction for selectively arranging data, specifying a data selection operand and a first and a second register each having a register width, the first and second registers providing a plurality of data elements each having an elemental width smaller than the register width, the data selection operand comprising a plurality of fields each independently selecting one of the plurality of data elements (emphasis added)**

The Office Action points to Blelloch's inverse-permute instruction as disclosing this claimed feature. *See* Office Action dated March 19, 2009, page 5, paragraph 8. In particular, the Office Action points to the indices of Blelloch's index vector (which is an input to the inverse-permute instruction) as supposedly disclosing the data selection operand recited in the claim.[2] However, Blelloch explicitly states that the all of indices of its vector index must be unique – *i.e.*, they must be different from one another:

> **As with the permute instruction, all indices must be unique. Blelloch, p. 66 (description of inverse-permute instruction) (*emphasis added*).**

Because the indices of Blelloch's vector index must all be unique (*i.e.*, different from one another), the selection of each index must take into account the values of all of the other indices, so as to not repeat the value of another index. This can be illustrated by the index vector disclosed by Blelloch, which is reproduced below (Blelloch at p. 66):

A	=	[a_0	a_1	a_2	a_3	a_4	a_5	a_6	a_7]
I	=	[3	0	7	2	6]			
`inverse-permute(A, I)`	=	[a_3	a_0	a_7	a_2	a_6]			

Here, the index vector is [3 0 7 2 6]. The first index is selected to have a value of 3. The selection of this value is dependent on the values selected for the other indices. Specifically, this index value cannot be 0, 7, 2, or 6, which are selected for the other indices. The same is true for the selection of every other index in the index vector. For instance, the second index is selected to have a value of 0. The selection of this value is again dependent on the values selected for the other indices. The second index value cannot be 3, 7, 2, or 6, which are selected for the other indices. The selection of the third, fourth, and fifth indices all have this

[2] The term "index vector" is used here, in order to be consistent with the Office Action. *See* Office Action dated March 19, 2009, page 5, paragraph 8 ("Each element in Index Vector I..."). Blelloch refers to this feature as the "indices vector." *See* Blelloch, p. 66, description of inverse-permute instruction ("The *values* vector must be equal or longer than the *indices* vector" (*emphasis in original*).

characteristic. That is, each index is selected by taking into account the values selected for the other indices, so as to not repeat any value selected for another index. Thus, the selection of each index in Blelloch is dependent on, not independent of, the values selected for all the other indices. Clearly, Blelloch's index vector fails to disclose a data selection operand comprising a plurality of fields each independently selecting one of a plurality of data elements, as required by claim 1. For at least this reason, Blelloch cannot anticipate the claimed invention.

2. Blelloch fails to disclose, in a single programmable processor, providing in parallel the data elements selected by the fields to respective predetermined positions in a catenated result

Blelloch fails to disclose, in a single programmable processor, providing in parallel the data elements selected by the fields to respective predetermined positions within a single programmable processor. The claimed invention is directed to the use of a single programmable processor to achieve parallel operation, and in particular, to the provision in parallel of data elements selected by fields of a data selection operand to respective predetermined positions in a catenated result. Claim 1 recites, in part:

> **A method of processing data in a single programmable processor, the method comprising**
>
> . . .
>
> **providing in parallel the data elements selected by the fields to respective predetermined positions in a catenated result, wherein the predetermined positions are in the same order as the fields of the data selection operand. (*emphasis added*)**

By contrast, Blelloch is directed to the use of multiple processors to achieve parallel operation, which was already known in the art. As Blelloch puts it: "The goal of this unification is to make it possible to program parallel algorithms in high-level languages, to have the algorithms execute efficiently *on a diverse set of real machines*" (emphasis added). Blelloch, p. 1, Introduction section, paragraph 2.

As for each individual processor within Blelloch's group of multiple processors, such a processor is merely a conventional scalar processor – *i.e.*, a processor that perform operations in

a purely sequential manner, not in parallel. In fact, Blelloch explicitly states that each individual processor "loops" over the elements in a vector that it is responsible for handling. As Blelloch states at p. 9, section 1.3, paragraph 2:

> **When executing a vector instruction on a vector *a*, each processor loops over the elements of *a* it is responsible for.**

In other words, each individual processor executes in a "loop" manner, performing the required operation on the first element, then looping back to perform the operation on the second element, then looping back to perform the operation on the third element, and so on. Such iterations in a "loop" execution are sequential in nature. Thus, each individual processor in Blelloch clearly performs operations sequentially, not in parallel.

Further evidence of the sequential nature of each individual processor is found in Blelloch's description of the amount of time it takes for a group of p processors to complete a vector operation of length n. Specifically, Blelloch states:

> **For a vector of length *n*, and for *p* processors, each primitive will run in *O(n/p+1)* time.**

That is, n operations performed by p processors requires $O(n/p+1)$ amount of time, where O is the amount of time required for one processor to complete one operation. Just as a simple example, if there are one thousand operations (n = 1000) to be performed by ten processors (p = 10), the amount of time required to complete all of the operations would be O * (1000/10+1), which is approximately O*100. Thus, using ten processors to perform these operations only improves the execution time by approximately a factor of ten (from 1000*O to 100*O). The improvement in execution time is simply the result of using more scalar processors that each performs one operation at a time – not the result of using a single processor that is capable of performing multiple operations in parallel. Clearly, each processor in Blelloch is simply

performing scalar operations in a conventional manner, *i.e.*, performing one operation at a time, not in parallel.[3]

Accordingly, claim 1 as amended distinguishes over Blelloch for at least the two separate reasons: (1) Blelloch fails to disclose a data selection operand comprising a plurality of fields each independently selecting one of a plurality of data elements; and (2) Blelloch fails to disclose, in a single programmable processor, providing in parallel the data elements selected by the fields to respective predetermined positions in a catenated result. For at least these reasons, Blelloch cannot anticipate claim 1.

The Office Action rejected claims 13, 14, 26, 40, 50, 54, and 64 as being anticipated by Blelloch, based on the similar rationale as claim 1. Correspondingly, Applicants respectfully submit that Blelloch fails to anticipate claims 13, 14, 26, 40, 50, 54, and 64 for similar reasons as stated above with respect to claim 1.

Claims 2, 4-8, and 11-12 depend from claim 1 and incorporate all of its limitations. Blelloch therefore fails to anticipate claims 2, 4-8, and 11-12 for at least the reasons stated above with respect to claim 1.

Claims 15, 17-21, and 24-25 depend from claim 14 and incorporate all of its limitations. Blelloch therefore fails to anticipate claims 15, 17-21, and 24-25 for at least the reasons stated above with respect to claim 14.

---

[3] Blelloch explicitly mentions the permute instruction (which is related to the inverse-permute instruction) as an example in this discussion of the execution time of a vector operation. Blelloch, page 9, section 1.3, paragraph 2 ("For a *permute* instruction, each processor reads a value and index for its elements, and writes the value in the destination vector at the position specified by the index... For a vector of length n, and for p processors, each primitive will run in $O(n/p+1)$ time"). While Blelloch does not explicitly mention the inverse-permute instruction in this discussion, all indications are that execution time for the inverse-permute instruction would be estimated in the same manner.

Claims 41, 43, and 46 depend from claim 40 and incorporate all of its limitations. Blelloch therefore fails to anticipate claims 41, 43, and 46 for at least the reasons stated above with respect to claim 40.

Claims 55, 57, and 60 depend from claim 54 and incorporate all of its limitations. Blelloch therefore fails to anticipate claims 55, 57, and 60 for at least the reasons stated above with respect to claim 54.

B. Lee

In section 21 of the Office Action, claims 1, 11, 14, and 24 were rejected under 35 U.S.C. § 102(e) as being anticipated by Lee (U.S. Patent No. 6,381,690). However, Lee is not prior art to the present application. In the prior response, Applicants submitted declarations under 37 C.F.R. § 1.131 from the inventors of the present application, Mr. Craig Hansen and Dr. John Moussouris, swearing behind Lee. The current Office Action has objected to these declarations, by pointing out that the documents provided in support of the inventor's declarations do not include sufficient details to establish conception and reduction to practice of the claimed invention prior to the effective date of the Lee reference. *See* Office Action, pages 3-4, paragraph 4.

In response, Applicants submit herewith a second, updated declaration under 37 C.F.R. § 1.131 from the Mr. Hansen. Mr. Hansen's second declaration is submitted with additional documentation not included in the prior declarations and further explains the documents that were submitted with the original declaration. The additional documentation together with the more complete explanation of all the documentation clearly provide sufficient details to establish conception and reduction to practice of the claim invention prior to the effective filing date of Lee.

Applicants respectfully submit that Mr. Hansen's second declaration and the documents cited therein provide sufficient additional evidence to establish that Lee indeed is not prior art to the present invention. Because Lee is not prior art, Applicants request withdrawal of the anticipation rejections based on Lee.

C. Lee in view of Matsuura

In section 24 of the Office Action, claims 12 and 25 were rejected under 35 U.S.C. § 103(a) as being unpatentable over Lee in view of Matsuura (U.S. Patent No. 4,725,973).

As discussed above, Applicants respectfully submit that Mr. Hansen's second declaration and the documents cited therein provide sufficient additional evidence to establish that Lee indeed is not prior art to the present invention. Applicants therefore request withdrawal of the rejection of claims 12 and 25 based on Lee in view of Matsuura.

D. Blelloch in view of *In re Rose*

In section 25 of the Office Action, claims 3, 9, 10, 16, 22, 23, 42, 44, 45, 47-49, 51-53, 56, 58, 59, 61-63, and 65-67 were rejected under 35 U.S.C. § 103(a) as being unpatentable over Blelloch in view of *In re Rose*, 105 U.S.P.Q. 237 (C.C.P.A. 1955).

Claims 3, 9, and 10 depend from claim 1 and incorporate all of its limitations. As discussed previously, Blelloch fails to disclose certain claim limitations of claim 1. The principles of *In re Rose* do not make up for the deficiencies of Blelloch with respect to such claim limitations. As such, even if Blelloch and *In re Rose* were combined, the resulting combination would still fail to disclose such claim limitations, which are incorporated into claims 3, 9, and 10. For at least this reason, claims 3, 9, and 10 are patentable over the combination of Blelloch and *In re Rose*.

Claims 16, 22, and 23 depend from claim 14 and incorporate all of its limitations. As discussed previously, Blelloch fails to disclose certain claim limitations of claim 14. The principles of *In re Rose* do not make up for the deficiencies of Blelloch with respect to such claim limitations. As such, even if Blelloch and *In re Rose* were combined, the resulting combination would still fail to disclose such claim limitations, which are incorporated into claims 16, 22, and 23. For at least this reason, claims 16, 22, and 23 are patentable over the combination of Blelloch and *In re Rose*.

Claims 42, 44, 45, and 47-49 depend from claim 40 and incorporate all of its limitations. As discussed previously, Blelloch fails to disclose certain claim limitations of claim 40. The principles of *In re Rose* do not make up for the deficiencies of Blelloch with respect to such claim limitations. As such, even if Blelloch and *In re Rose* were combined, the resulting combination would still fail to disclose such claim limitations, which are incorporated into claims 42, 44, 45, and 47-49. For at least this reason, claims 42, 44, 45, and 47-49 are patentable over the combination of Blelloch and *In re Rose*.

Claims 51-53 depend from claim 50 and incorporate all of its limitations. As discussed previously, Blelloch fails to disclose certain claim limitations of claim 50. The principles of *In re Rose* do not make up for the deficiencies of Blelloch with respect to such claim limitations. As such, even if Blelloch and *In re Rose* were combined, the resulting combination would still fail to disclose such claim limitations, which are incorporated into claims 51-53. For at least this reason, claims 51-53 are patentable over the combination of Blelloch and *In re Rose*.

Claims 56, 58, 59, and 61-63 depend from claim 54 and incorporate all of its limitations. As discussed previously, Blelloch fails to disclose certain claim limitations of claim 54. The principles of *In re Rose* do not make up for the deficiencies of Blelloch with respect to such

claim limitations. As such, even if Blelloch and *In re Rose* were combined, the resulting combination would still fail to disclose such claim limitations, which are incorporated into claims 56, 58, 59, and 61-63. For at least this reason, claims 56, 58, 59, and 61-63 are patentable over the combination of Blelloch and *In re Rose*.

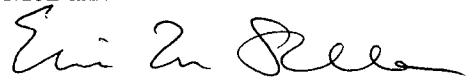
IV. CONCLUSION

In view of the foregoing, Applicants submit that all claims now pending in this Application are in condition for allowance. The issuance of a formal Notice of Allowance at an early date is respectfully requested. If the Examiner believes a telephone conference would expedite prosecution of this application, please telephone the undersigned at telephone number indicated below.

To the extent necessary, a petition for an extension of time under 37 C.F.R. 1.136 is hereby made. Please charge any shortage in fees due in connection with the filing of this paper, including extension of time fees, to Deposit Account 500417 and please credit any excess fees to such deposit account.

Respectfully submitted,

McDERMOTT WILL & EMERY LLP



Eric M. Shelton
Registration No. 57,630

Please recognize our Customer No. 20277 as our correspondence address.

600 13th Street, N.W.
Washington, DC 20005-3096
Phone: 202.756.8000 EMS:MWE
Facsimile: 202.756.8087
Date: September 21, 2009